Predicting particle accelerator failures using binary classifiers

Miha Rescic[a,1,*], Rebecca Seviour[a], Willem Blokland[b]

[a] *University of Huddersfield, Queensgate, Huddersfield, HD1 3DH, UK*
[b] *Neutron Sciences Directorate, One Bethel Valley Rd, Oak Ridge, TN 37831, USA*

Abstract

Particle accelerator failures lead to unscheduled downtime and lower reliability. Although simple to mitigate while they are actually happening such failures are difficult to predict or identify beforehand. In this work we propose using machine learning approaches to predict machine failures via beam current measurements before they actual occur. To demonstrate this technique in this paper we examine beam pulses from the Oakridge Spallation Neutron Source (SNS). By evaluating a pulse against a set of common classification techniques we show that accelerator failure can be identified prior to actually failing with almost 80% accuracy. We also show that tuning classifier parameters and using pulse properties for refining datasets can further lead to almost 92% accuracy in classification of bad pulses. Most importantly, in the paper we establish there is information about the failure encoded in the pulses prior to it, so we also present a list of feasible next steps for increasing pulse classification accuracy.

Keywords: particle accelerator, accelerator reliability, machine learning, binary classifier, failure prediction

1. Introduction

Reliability analysis and reliability modeling have been widely applied in many engineering disciplines, from civil engineering [1] to aerospace [2] and,

*Corresponding author
Email addresses: miha.rescic@hud.ac.uk (Miha Rescic), R.Seviour2@hud.ac.uk (Rebecca Seviour), blokland@ornl.gov (Willem Blokland)

in last two decades, to particle accelerators as part of an Accelerator Driven System (ADS) [3, 4]. ADS require high reliability and availability which restrict accelerator downtime to few seconds on a daily scale [5, 6]. Such requirements are currently beyond the performance of existing machines today [7–13].

Historically, reliability analysis for accelerators was usually performed in the design phase [14, 15]. Some of the methods in the first case include: Reliability Block Diagrams (RBD) [4, 16], Fault Tree Analysis (FTA) [17–19], Failure Mode and Effect Analysis (FMEA) [20] and other methods [21, 22]. The results of these methods are later validated by analyzing machine operations data [9, 23]. Recently, more methods to actively address reliability have been proposed [24–26]. Common to all methods is the dependency on underlying reliability data which in accelerator field is sparse [3].

As a field, machine learning has evolved out of advances of artificial intelligence research, especially computational learning theory and pattern recognition. The term was first established by IBM's Arthur Samual in 1959 with the idea to give *computers the ability to learn without being explicitly programmed* [27]. The field itself contains a vast body of knowledge covered by many papers and books, [28–31] being some of them.

Within the machine learning area the field of pattern recognition is the one we are interested in. The area has been growing and advancing since the 1960s [32, 33] and seen increase in pace the last 20 years due to advances in methods [34–38] and increase of computing power and data available [39–41]. Due to the nature of the data available to us we have focused specifically on two classification techniques: k-Nearest Neighbors (kNN) and Decision Trees (DT).

The application of machine learning to particle accelerator was first applied in the 1980s [42] with the first use of neural networks in 1989 [43] and the application in 1991 [44] for a dynamic feedback system for beamline control. Recently there is growing interest in the applications of machine learning techniques to particle accelerators, e.g research into using machine learning techniques for particle accelerator control [45], RF gun temperature control [46], RF pulses shape [47], LLRF feedforward control [48] and for predicting disruptions in operations

of fusion reactors [26]. Most recent workshops include topics like predictive control of particle accelerators [49], detecting faulty beam position monitors [50, 51], tuning XFEL machine using machine learning [52] and predicting and counteracting machine interlocks [53].

2. Classification problem description

2.1. Background

During the operation of the accelerator the machine protection system (MPS) sometimes shuts down the machine to prevent damage. This happens when the beam in the accelerator enters an off-normal state (e.g. the position is off by more than a certain threshold) which leads to the MPS interlocks to stop the generation of the particle pulses.

Such machine trips lead to the acquisition and storage of the beam current in three different time positions relative to the machine trip. The first stored pulse waveform is the last pulse that successfully passed through the accelerator, namely the *previous* pulse to the trip one. The next measurement saved is the actual trip pulse causing the interlock. The last pulse that is stored in the collection is the first successive pulse that passes successfully through the accelerator after it's restarted, namely the *next pulse*. We consider the latter as a signal that the accelerator has entered a normal state again.

Our research is focused on labeling *next* pulses with the label *good* - the classified pulse will not lead to a machine trip and the accelerator will remain in normal operating state and *previous* pulses with the label *bad* - the machine will trip on the pulse following the *bad* pulse and normal operation will be interrupted.

2.2. Changing accelerator and malfunctioning equipment

We need to note that the accelerator is in the process of changing all the time. Most of bad pulses are caused by malfunctioning equipment, which leads to the equipment being fixed or replaced, which leads to bad pulses being generated by another malfunctioning equipment etc. This means the specific failure patterns

of the waveforms in datasets are changing. For example, during the time in 2015 when the data in our dataset was acquired, SNS had between 200 to 1000 bad pulses per day, currently, in 2019, during a good day the number is around 20.

We should also note that more than one piece of equipment can be malfunctioning at the same time. Such interleaved failure combinations can be very specific for a certain point in time. They can also have very specific patterns but are hard to map back to the source equipment.

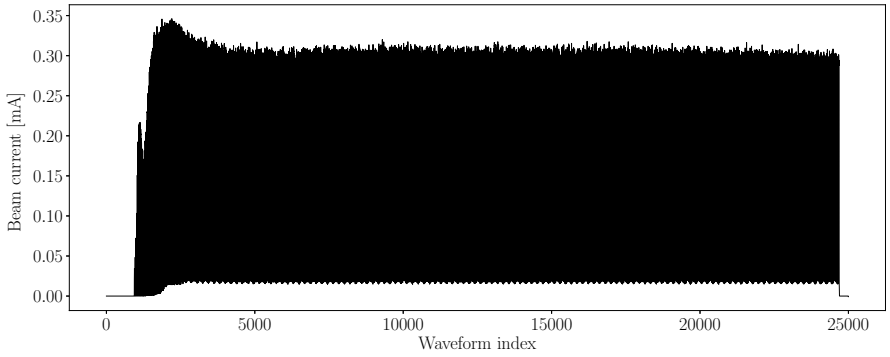
The above means two things. First, that every waveform in the dataset has a very specific underlying accelerator configuration signature at the acquisition time. Therefore, the dataset is continuously changing with time. Second, precursors have been found where a specific equipment malfunction caused specific failures. This means that specific failure patterns can be linked to specific equipment failure.

For the purpose of the paper, we take a holistic approach to failure identification not using any information about specific equipment failure. We also use an uniform approach to the data not taking the above mentioned changes in time into account. We acknowledge that both taking different system failures into account and splitting the dataset further is feasible, but introduces too many unknowns to serve as a solid starting point.

2.3. Dataset and problem

Our approach is based on observing the raw SNS beam current data. The data is measured by a differential beam current monitor [54] that has sensors at two locations (*upstream* - closer to the particle source and *downstream* - further away from the particle source). The measured difference is used to abort beam immediately if there is a difference between the upstream and downstream measurement, to minimize the beam loss. Figure 1 shows a waveform for a beam current measurement. We have run the analysis with data from both locations and there was no significant difference in received results. Therefore, for the purpose of this paper we use the samples acquired from the *upstream* beam

Figure 1: Single pulse beam current waveform



current monitor.

The waveforms are composed of 25000 data points each and our dataset is composed of 15 different days worth of measurements. The data used in this study was collected on days of normal machine operation - the goal for the machine was to maintain a steady power on target (as compared to beam study operations where the purposes are studying various properties of the machine under different operational parameters).

The dataset contains approximately 15000 waveforms, of which half are *good* pulses and half are *bad* pulses. The ratio 50/50 comes from the limitations of the data acquisition system (one pulse of each type acquired per machine trip). The dataset itself is unique as SNS is the only facility today recording information directly prior to the failure.

Within the classification problem there are two scenarios we might find ourselves in: either we posses the full statistical knowledge of the statistical distribution of the observation x and the category σ or we do not have any knowledge of the underlying distribution and we must derive one from the samples that are available to us.

We find ourselves in the first scenario when we look at our sample data - we know exactly which *distribution* generated the sample and which category it is: *pretrip*, *trip* or *normal*. This is because the classification was done by

the MPS system after the event occurred. The MPS system, after accelerator recovers, stores and labels the three pulses related to event. What our method proposes to address is the second scenario - where we're given a live pulse from the accelerator without the correct classification (we do not know if the current, the next or the n^{th} pulse will trip the machine) and we can only derive it's category from the N samples we've already acquired and categorized.

The waveforms in our dataset are assigned labels based on how the data acquisition system acquired and stored them: *previous* pulses are assigned a *bad* label and *next* pulses are assigned a *good* label.

In this section we introduce a set of classifiers we've chosen for the evaluation. The primary motivation for classifier selection was both applicability to our use case (large feature set, a binary classifier) and availability of libraries. Therefore, we have used python and scikit-learn libraries to perform classification and benchmarking.

3. Methods

3.1. k Nearest Neighbors

In pattern recognition, the k-nearest neighbors algorithm (kNN) is a non-parametric method used for classification introduced first in 1951 and developed and researched in the later years [55–57].

The category of an un-classified sample s is determined as the most heavily representative category among the previously classified k neighbors, which exist as a subset in a much larger space of N previously classified samples. The distance between samples s_i and s_j of length n is calculated using the Euclidean norm defined in equation 1.

$$d(s_i, s_j) = \sqrt{\sum_{k=1}^{n} (s_{ik} - s_{jk})^2} \tag{1}$$

In practice it turns out that adding weights [58] to sample distances improves the classification precision and there have been many other improvements and applications in the years since the first introduction of the algorithm [59–63].

The most important tunable paremeter is k - the number of neighbours to take into account when assigning the label to a sample. We explore different values of k and the effects on classification performance in Section 4 and Figure 6.

3.2. Decision Tree classifiers and regressors, Random forests and Gradient Boost

Another method from the classification field is the Decision Tree (DT) classifier. Trees are directed graphs beginning with one node and branching to many. They are fundamental to computer science as data structures and many other fields. Decision tree method uses and traverses a tree-like data structure to go from observations stored and represented in the branches to conclusions about the target classification represented in the tree leaves [64–66]. We use the tree type labeled *classification trees* [64], where the input is a discrete set of variables. In these tree structures, leaves represent class labels and branches represent the set of features and their values that lead to the chosen class labels.

Generalization of the decision tree method is an ensemble classifier named Random forests classifier [67] where a set of decision tree classifiers is generated and the classification is performed by each of the trees casting a vote.

Similar to a decision tree is a Regression tree [64], where instead of a discrete output of the classification process (the sample predicted class) the desired output is numeric or continuous, e.g. predicted future price of certain goods as compared to input sample class.

Gradient boost is an ensemble classifier that uses sets of regression trees together with majority voting for sample classification [68, 69].

Both Random Forest and Gradient boost are build on top of decision trees, so the most relevant tunable hyperparameter is the number of classifiers (decision trees) voting on the assigned label. Effects on using different values on performance of classifiers are presented. in Section 4 and Figure 6

3.3. (Linear) support vector machines

Support Vector Machines (SVM) are a set of supervised learning methods used for classification [70]. Training is performed by defining separation hyper-

planes (e.g. lines on a 2D surface) between training samples to best classify the training set. Classification is then performed by observing where the samples' position is and which side of the hyperplane(s) it's positioned at.

In code, the model is represented by a matrix (kernel) and the classification is performed by multiplying the input sample vector with the kernel matrix, the obtained result is a vector of predicted sample classes.

Linear support vector classifier is a version of the support vector machine classifier where the classification kernel is linear.

3.4. Perceptrons and Neural networks

The perceptron [71] is a simple linear classifier that given an input sample, using a threshold function, determines the sample classification. Training is performed by tuning the function hyperparameters to best match the training datasets sample classes.

A single perceptron is also considered a single layer neural network [72]. More complex neural networks are composed of multiple layers of perceptrons with connected inputs and outputs.

3.5. Other

Logistic regression [73] classifier uses a logistic function as the model for classification of samples.

Naive Bayes classifiers [74] are a set of supervised learning algorithms based on Bayes' theorem with the "naive" assumption of conditional independence between every pair of sample features, given the value of the class variable.

4. Results

In this section we present the results of our analysis using the following four metrics:

- *Train precision* - the ability to apply the correct label to the correct class on the training data,

- *Test precision* - the ability to apply the correct label to the correct class on the test data,
- *True positive* - the ability to apply the *good* label the *good* pulse,
- *True negative* - the ability to apply the *bad* label the *bad* pulse.

The latter entity is the main focus of our research since identifying a bad pulse has a higher success than identifying a good pulse. This is due to the fact that a running accelerator machine will have many more good pulses in a given time frame than bad pulses.

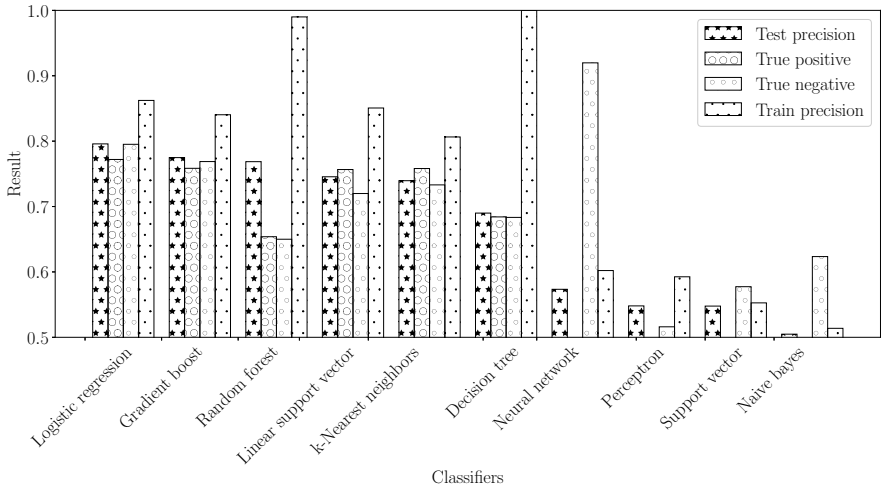
For example, the accelerator would ideally, without failures, generate about 5×10^6 pulses in a day (a 60 Hz machine running for 24 hours). Discounting for average accelerator availability of 92% during years 2011-2013 [11] we can estimate the the average number of pulses per day of operations to be around 4.7×10^6.

From our dataset we can observe that that the SNS accelerator can have anything between 200 and more than 1000 bad pulses during a normal operating day, so let's average to 500 bad pulses on average per day. This means that the number of *False negatives* (good pulses identified as bad) would be, assuming 80% classification success, around 9×10^5. Alternatively, same classification success would yield only 100 *False positives* i.e. bad pulses identified as good.

First, we need to note that it is possible to abort beam 9×10^5 times per day, but it would lead to users being disapointed with the reduced amount of neutrons due to less power on target. Second, we also need to note that currently, the MPS system has a rate of 100% False positives, since all bad pulses are labeled good as SNS permits them.

We present all four values in our graphs because, although focused on TNs, a classifiers performance cannot be evaluated only from that value. For example, a very bad classifier that would just label all samples as bad would still have 100% precision for TN, but a 50% test precision, which would be a signal that something might be off with the classifier. There is also other information one can get from the other values, e.g. *train precision* compared to *test precision*

Figure 2: Classifier performance for the whole dataset, ranked best to worse by test accuracy



will tell us how much the model is *overfitted* - trained too closely and exact to the trained data leading leading to poor test results.

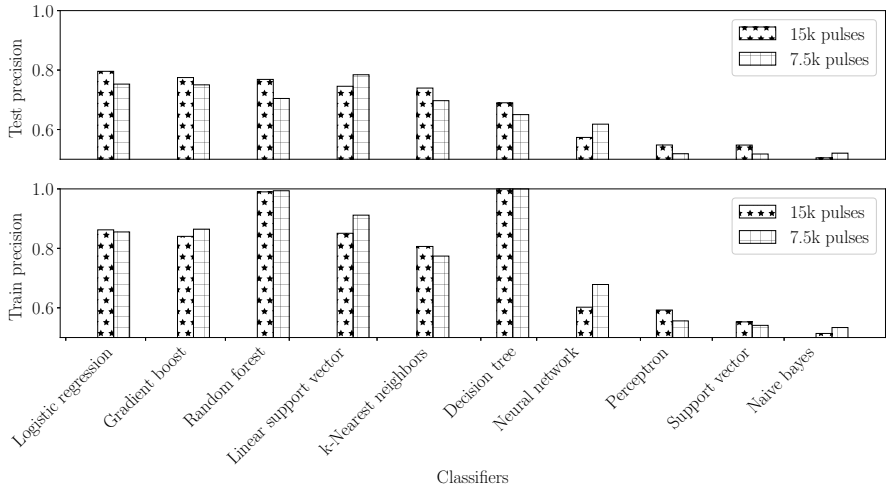
As the first step we have evaluated the performance of a collection of classifiers available from the libraries. Most of the classifiers were configured using default parameters (except k-nearest neighbors where the value $k = 11$ was used instead of default $k = 5$). In the first iteration the complete dataset was used (15000 samples) and a 5-fold exhaustive cross-validation was used to measure results. All values are the mean of all five folds.

Additionally, we also measure *train precision* which is a measure of how accurate are the trained models on the training data. This measure compared to *test precision* will tell us how much the model is *overfitted* - trained too closely and exact to the trained data leading leading to poor test results.

The results of the benchmark, sorted by descending test accuracy are presented in Figure 2. From the results we can observe the following:

- Logistic regression classifier is the best performing with 79.5% test precision

Figure 3: Classifier test and train performance for the complete dataset and a subset of randonly selected samples



- Both Random forest and Decision tree are heavily overfitted on the test data
- Neural network overall precision is hardly above the 50% threshold but the True negative score is the best of all classifiers at 91.9%

With machine learning it's common that the size of the dataset affects the quality of classification, so the second evaluation measures the classification scores as a function of dataset size. This is of our limitations since the pulses are only acquired on machine trips and the number of those has been reducing over the last few years. In addition, we are also limited by the computation time needed to perform these classifications. We compared the classifier performance on the complete dataset (15000 pulses) and a dataset of half the size (7500 randomly selected pulses).

Figure 3 shows test and train precision as mentioned before in Figure 2 but split to the two datasets of different size. We can observe from the graphs:

- Both Random forest and decision tree are overfitted on the smaller dataset

Figure 4: Performance of kNN as a function of k

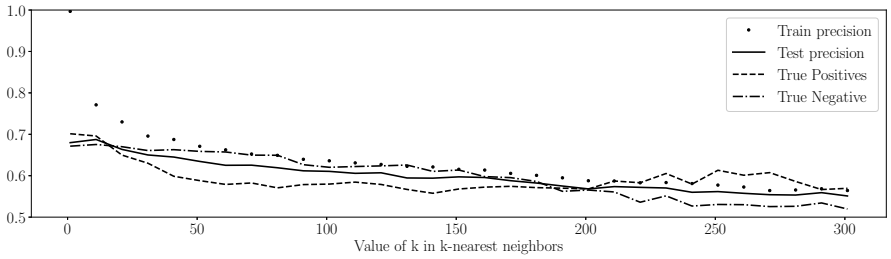
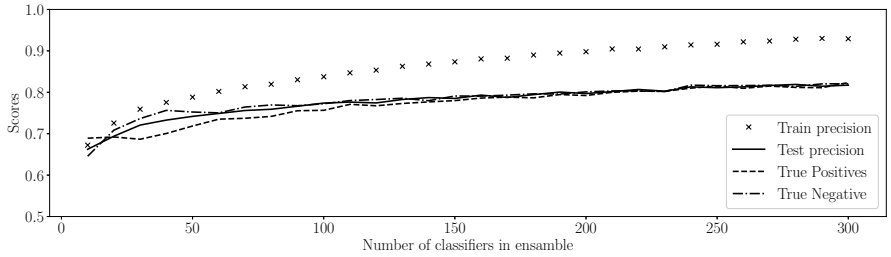


Figure 5: Performance of Gradient boost classifier as a function of number of estimators in ensamble

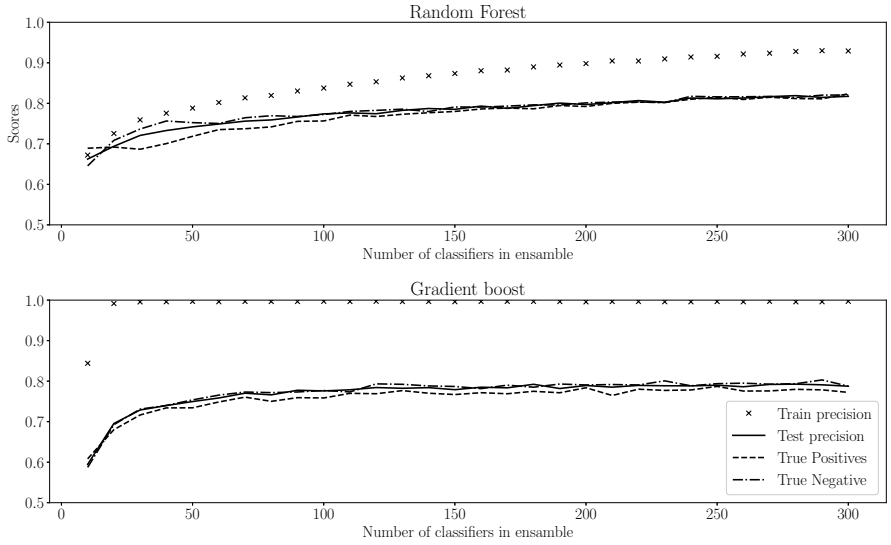


too

- Classifier performance does change (although not drastically) as the dataset size doubles: best performing classifiers Linear regression and Gradient boost (79,5% and 76,9% True Negative respectively) increase in performance by 3,7% and 2,4% respectively as the dataset size doubles
- Neural network actually performs worse on the bigger dataset

Seeing that increase in dataset size will not lead to significant classification improvements, we took the next step to tune the main classifier hyperparameters. First, we looked at the number of k neighbors in the kNN classifier. Results are presented in Figure 4 and are very typical for kNN classifier: the larger the setting, the worse are the results due to too many neighbours taken into account and blurring the line between the two classes. Second, we looked

Figure 6: Performance of Random forest and Gradient boost classifiers as a function of number of estimators in ensamble



at the number of classifiers in the Random Forest and Gradien Boost classifier. Results are shown in Figure 6; the figure tells us that the more estimators we take, the better the classification results even if the classifiers were overfit on the training set.

The top five best performing classifiers have different results when it comes to precision so joining them together in a voting classifier could increase performance. A majority rule was applied for the label votes. Figure 7 shows the results of the benchmark where we can observe that no significant improvement was achieved.

The last benchmark was performed splitting the dataset into 10 smaller datasets using individual pulse raw data variance. We chose pulse variance as a start since it is affected by many beam and accelerator parameters, most relevant being: power on target (amplitude), bunch length (length of peaks in the waveform) and bunch count (number of peaks in waveform). Figure 8 shows how sample variance differs on different acqusition days.

Figure 7: Performance of voting classifiers compiled of up to five top performing classifiers

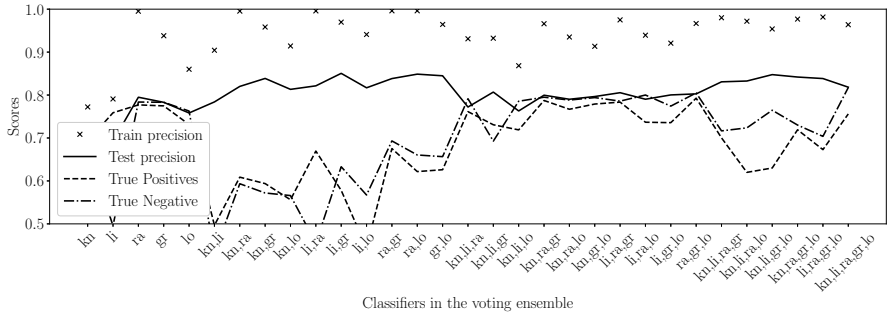


Figure 8: Minimum, average and maximum sample variance per acquisition day

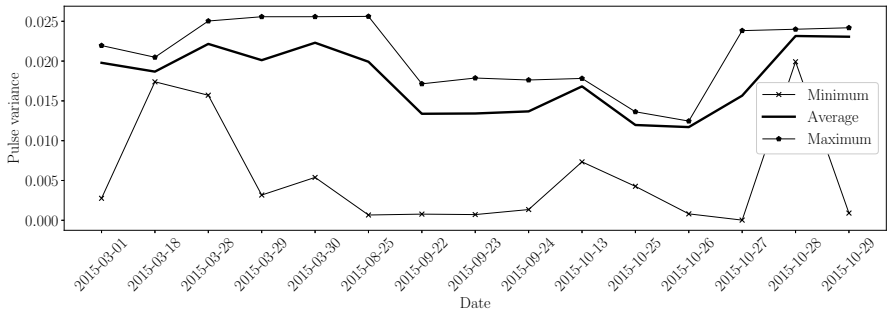
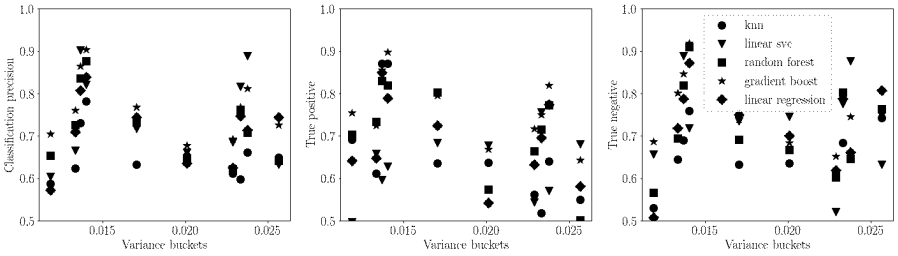


Figure 9: Performance of classifiers on data subset split by sample variance



This resulted in 10 datasets of 750 pulses on which the top 5 classifiers were evaluated using the best performing hyperparameters from Figures 4 and 6 (10 k-neighbours and 300 estimators in enseble for RF and GB). The results of the last benchmark are presented in Figure 9. The bin ranges were set so the number of pulses in each bin would be approximately the same (about 1500 per bin) so some bins are groupped closer than others.

The observation we can make from the results is the following: classifiers can perform better or worse on the smaller datasets compared to the benchmark on the complete dataset presented in Figure 2. Examining Logistic regression (LR) and Gradient boost (GB) as the two best performing (benchmark of 79.5% and 76.9% True Negative respectively) we can observe that LR performs better than benchmark on 2 buckets (87.2% and 80.7%), close to benchmark on 3 buckets (78.7%, 78.8% and 75.0% and worse than benchmark on the remaining 5 buckets (performance dropping almost to 50% on one of the buckets). The results for GB are: 4 buckets outperform benchmark (91.2%, 84.7%, 80.2% and 78.7%), 3 are comparable to benchmark (75.1%, 74.5% and 73.9%) whereas remaining three are lower than benchmark (worse being 65.5%). One also notes that the over and under performing buckets are not the same for LR an GB.

5. Conclusions

In the paper we have presented that we can achieve solid (79.5% on a generalized dataset and up to 91.9% on a metadata grouped dataset) correct True Negative (bad pulse labeled as bad) classification of pulses. Using the example

from the beginning of section 4 the mentioned accuracy yields the following results: around $3,8 \times 10^5$ False Negatives (good pulses labeled as bad) pulses (8% of average daily pulse count) and around 40 False Positives (bad pulses labeled as good) pulses on a daily basis.

Such high number of False Negatives mean the current model performance is unsuitable for reliable implementation of a mitigation system. This is because mitigating all the detected False Negatives would lower the beam power on target. Regardless of current performance, most importantly, we have confirmed there is distinct information about future failures present in non-failing pulses prior to the actual failure and offer a solid reference for future research.

We have shown that an increase of the dataset size does lead to increased classifier accuracy, although two orders of magnitude less than the actual dataset size increase (doubling the dataset size leads to 3.7% and 2.4% increase in performance for the two best performing classifiers. Therefore, increase of the dataset size is the first proposed step for future research.

Although not directly addressed in the paper, we believe that dimensionality reduction of samples (using e.g. Principal Component Analysis) should also be investigated. We recommend using less but more representative information per sample, compared to the current approach of using raw data (25000 datapoints per sample).

Lastly, we have shown that focusing on grooming datasets based on sample metadata (we explored sample variance) also leads to the best classifier performance improvements (7.7% and 14.4% improvement for the two best classifiers respectively). This point is very important when addressing the changes in the accelerator configuration (see Section 2.2). To use historical data for future predictions, the current state of the accelerator at a given time must be described with as much metadata as possible. Therefore, we believe more metadata (data as beam power on target, machine uptime, pulse bunch count etc.) must be acquired and included in the datasets.

References

[1] M.H. Faber and M.G. Stewart. Risk assessment for civil engineering facilities: critical overview and discussion. In *Reliability Engineering & System Safety*, volume 80, pages 173–184. 2003.

[2] H.J. Pradlwarter, M.F. Pellissetti, C.a. Schenk, G.I. Schuëller, a. Kreis, S. Fransen, a. Calvi, and M. Klein. Realistic and efficient reliability estimation for aerospace structures. In *Computer Methods in Applied Mechanics and Engineering*, volume 194, pages 1597–1617. 2005.

[3] P Pierini. Reliability studies of the PDS-XADS accelerator. *PDS-XADS deliverable 57: Potential for Reliability Improvement and Cost Optimization of Linac and Cyclotron Accelerators*, 2003.

[4] Paolo Pierini and Luciano Burgazzi. Ads accelerator reliability activities in europe. *Utilisation and Reliability of High Power Proton Accelerators: Workshop Proceedings*, May 2004.

[5] Nuclear Energy Agency. Organisation for Economic Co-operation and Development. NEA Accelerator-driven Systems (ADS) and Fast Reactors (FR) in Advanced Nuclear Fuel Cycles: A Comparative Study, 2002.

[6] S. Henderson et al. Accelerator and Target Technology for Accelerator Driven Transmutation and Energy Production White Paper Working Group Report. 2010.

[7] G. S. Bauer. SINQ Status report. In *Proceedings, 15th Meeting of the International Collaboration on. Advanced Neutron Sources (ICANS), November 6-9,2000. Tsukuba, Japan*, pages 103–107, 2000.

[8] F. Groeschel, S. Dementjev, H. Heyck, W. Leung, K. Thomson, W. Wagner, and L. Zanini. MEGAPIE Irradiation Experience of the First Megawatt Liquid Metal Spallation Target. In *Proceedings, 5th International Workshop on the Utilization and Reliability of High Power Proton Accelerators (HPPA5), May 6-9 2007, Mol, Belgium*, 2007.

[9] John Galambos, SH Kim, and I Campisi. SNS Experience with a High-Energy Superconducting Proton Linac. In *CARE-HHH-APD Event BEAM'07, Finalizing the Roadmap for the Upgrade of the LHC and GSI Accelerator Complex, 1-5 October 2007, CERN, Switzerland*, 2007.

[10] John Galambos. SNS Operational Experience at the MW Level SNS Operational Experience. In *Proceedings, 46th ICFA Advanced Beam Dynamics Workshop on High-Intensity and High-Brightness Hadron Beams (HB2010), September 27-October 1, 2010, Morschach, Switzerland*. PSI, 2010.

[11] John Galambos. SNS Performance and the Next Generation of High Power Accelerators. In *Proceedings, North American Particle Accelerator Conference (NAPAC), September 29-October 4 2013, Pasadena, California, USA*, pages 1443–1447, 2013.

[12] S H Kim, R Afanador, W Blokland, M Champion, A Coleman, M Crofford, B Degraff, M Doleans, D Douglas, T Gorlov, B Hannah, M Howell, Y Kang, S-w Lee, C Mcmahan, T Neustadt, S Ottaway, C Peters, J Saunders, A Shishlo, S Stewart, W H Strong, D Vandygriff, and Oak Ridge. The Status of the Superconducting Linac and SRF Activities at the SNS. In *Proceedings, 16th International Conference on RF Superconductivity, September 23-27, Paris, France*, pages 83–88, 2013.

[13] I E Campisi, S Assadi, F Casagrande, M Crofford, G Dodson, J Galambos, M Giannella, M Howell, Y Kang, K Kasemir, S H Kim, Z Kursun, P Ladd, H Ma, D Stout, Y Zhang, Oak Ridge, and M Champion. Status and Performance of the Spallation Neutron Source Superconducting Linac. In *Proceedings, 22nd Particle Accelerator Conference (PAC), June 25-29, Albequerque, New Mexico, USA, 2007*, pages 2502–2504, 2007.

[14] Luciano Burgazzi and Paolo Pierini. Reliability studies of a high-power proton accelerator for accelerator-driven system applications for nuclear

waste transmutation. *Reliability Engineering & System Safety*, 92(4):449–463, 2007.

[15] N Pichoff and H Safa. Reliability of superconducting cavities in a high power proton linac. *7th EPAC conference, Vienna*, pages 2049–2051, 2000.

[16] M Modarres, M Kaminskiy, and V Krivtsov. *Reliability engineering and risk analysis: a practical guide*. CRC Press, 2011.

[17] W E Vesely, F F Goldberg, N H Roberts, and D F Haasl. *Fault Tree Handbook*. Number NUREG-0492. Division of Systems and Reliability Research, Office of Nuclear Regulatory Research, U.S. Nuclear Regulatory Commission, 1981.

[18] Carlos Tapia, Javier Dies, Vicente Pesudo, Javier Abal, Ángel Ibarra, and José M. Arroyo. IFMIF accelerator: Database, FMEA, fault tree and RAM. In *Fusion Engineering and Design*, volume 86, pages 2722–2725. 2011.

[19] A E Pitigoi, P Fernández Ramos, Empresarios Agrupados, and Empresarios Agrupados Ea. Modelling High-power Accelerators Reliability - Reliability Model of SNS Linac. In *Proceedings, 39. Annual Meeting of Spanish Nuclear Society; 39. Reunion Anual Sociedad Nuclear Espanola, 25-27 September, Reus, Tarragona, Spain, 2013*, 2011.

[20] D H Stamatis. *Failure mode and effect analysis: FMEA from theory to execution*, volume 38. 2003.

[21] Enric Bargalló, Pere Joan Sureda, Jose Manuel Arroyo, Javier Abal, Alfredo De Blas, Javier Dies, Carlos Tapia, Joaquín Mollá, and Ángel Ibarra. Availability simulation software adaptation to the IFMIF accelerator facility RAMI analyses. In *Fusion Engineering and Design*, volume 89, pages 2425–2429. Elsevier B.V., 2014.

[22] N Phinney, T Himel, and MC Ross. Reliability Simulations for a Linear Collider. In *9th European Particle Accelerator Conference, 5 - 9 July, Lucerne, Switzerland, 2004*, pages 857–859, 2004.

[23] S Suhring. Accelerator availability and reliability issues. In *Proceedings, 2003 Particle Accelerator Conference (PAC), May 12-16, Portland, Oregon USA*, pages 625–629, 2003.

[24] T Tajima, M Borden, F Olivas, J Chamberlin, J Harrison, M Oothoudt, and A Canabal. Lansce Vacuum System Refurbishment Plan And Vacuum Alert System Improvements For Predictive Maintenance. pages 3717–3719, 2008.

[25] K Žagar, D Bokal, K Strniša, and M Gašperin. Predictive diagnostics for high-availability accelerators. In *Proceedings, International Particle Accelerator Conference (IPAC'13), Shanghai, China, May 12-17, 2013*, pages 873–875, 2013. ISBN 9783954501229.

[26] William Tang, Matthew Parsons, and Eliot Feibush. Machine Learning Studies of JET Disruption. 2015.

[27] Arthur L Samuel. Some Studies in Machine Learning Using the Game of Checkers. 3(3):535–554, 1959.

[28] Tom Mitchell. *Machine Learning*. Mcgraw Hill, 1997.

[29] Peter Bruce and Andrew Bruce. *Practical Statistics for Data Scientists*. O'Reily Media, 2017.

[30] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer, 2017.

[31] Yaser S. Abu-Mostafa, Malik Magdon-Ismail, and Hsuan-Tien Lin. *Learning From Data*. AMLBook, 2012.

[32] Lawrence Gilman. Roberts. *"Machine Perception of Three-dimensional Solids."*. Massachusetts Institute of Technology, 1963.

[33] David G Lowe. Three-Dimensional Object Recognition from Single Two-Dimensional Images. 3(March 1987):1–39.

[34] Paul Viola. Rapid Object Detection using a Boosted Cascade of Simple Features. 2001.

[35] David G Lowe. Distinctive Image Features from Scale-Invariant Keypoints. pages 1–28, 2004.

[36] Svetlana Lazebnik, Cordelia Schmid, Jean Ponce, Svetlana Lazebnik, Cordelia Schmid, and Jean Ponce. Beyond bags of features : spatial pyramid matching for recognizing natural scene categories. 2010.

[37] Pedro Felzenszwalb, David Mcallester, Deva Ramanan, and U C Irvine. A Discriminatively Trained , Multiscale , Deformable Part Model.

[38] Navneet Dalal, Bill Triggs, and De Europe. Histograms of Oriented Gradients for Human Detection. 2005.

[39] Jia Deng, Wei Dong, Richard Socher, Li-jia Li, Kai Li, and Li Fei-fei. ImageNet : A Large-Scale Hierarchical Image Database. pages 248–255, 2009. URL `http://www.image-net.org/challenges/LSVRC/`.

[40] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, C V Jan, J Krause, and S Ma. ImageNet Large Scale Visual Recognition Challenge. URL `http://www.image-net.org/challenges/LSVRC/`.

[41] Mark Everingham, Luc Van Gool, Christopher K I Williams, John Winn, and Andrew Zisserman. The PASCAL Visual Object Classes (VOC) Challenge. URL `http://host.robots.ox.ac.uk/pascal/VOC/`.

[42] T Higot, H Shoaee, and Stanford Linear Accelerator. Some Applications of AI to the Problems of Accelerator Physics. pages 701–703, 1987.

[43] A Corneliusen P Terdali T Knight and J Spencer. Computation and Control with Neural Nets. 1989, 1989.

[44] M Lee, R Sass, and H Shoaee. Accelerator and feedback control using neural networks. 1991:1–3, 1991.

[45] A L Edelen, S G Biedron, B E Chase, D Edstrom Jr, S V Milton, and P Stabile. Neural Networks for Modeling and Control of Particle Accelerators. 63(2):878–897, 2016.

[46] A L Edelen, S G Biedron, S V Milton, Fort Collins, B E Chase, D J Crawford, N Eddy, D Edstrom Jr, E R Harms, J Ruan, J K Santucci, and P Stabile. Initial experimental results of a machine learning-based temperature control system for an rf gun. pages 1217–1219.

[47] Amin Rezaeizadeh, Paul Scherrer Institut, Thomas Schilcher, Paul Scherrer Institut, and Roy Smith. Rf pulse flattening in the swissfel test facility based on model-free iterative learning control. pages 824–827.

[48] M Laverty and K Fong. An iterative learning feedforward controller for the triumf e-linac. pages 485–487.

[49] Auralee Edelen, Sandra Biedron, Daniel Bowring, Brian Chase, David Douglas, Jonathan Edelen, and Chip Edstrom. Experience with Model Predictive Control and Model-Based Reinforcement Learning. In *Machine Learning Applications for Particle Accelerators*, 2018.

[50] E Fol and R Tomas Garcia. Detection of faulty Beam Position Monitors. In *Machine Learning Applications for Particle Accelerators*, 2018.

[51] E Fol, F Carlier, A Garcia-tabares Valdivieso, and R Tomás. Machine Learning Methods for Optics Measurements and Corrections At LHC. *Proc of IPAC 2018*, pages 12–15, 2018.

[52] Alvaro Sanchez-gonzalez. Machine learning applied to single-shot x-ray diagnostics in an XFEL. 8, 2018.

[53] Andreas Adelmann and Jochem Snuverink. Forecasting of Beam Interlocks in High Intensity (Hadron) Accelerators. In *Machine Learning Applications for Particle Accelerators*, 2018.

[54] W Blokland and C Peters. A New differential and Errant Beam Current Monitor for the SNS Accelerator. *submitted to International Beam ...*, pages 921–924, 2013. URL `http://accelconf.web.cern.ch/AccelConf/IBIC2013/papers/thal2.pdf`.

[55] E Fix and JL Hodges. Discriminatory analysis, non-parametric discrimination. *DTIC Document*, (May), 1951.

[56] T. Cover and P. Hart. Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, 13(1):21–27, 1967.

[57] N. S. Altman. An introduction to kernel and nearest-neighbor nonparametric regression. *American Statistician*, 46(3):175–185, 1992.

[58] Sahibsingh A. Dudani. The Distance-Weighted k-Nearest-Neighbor Rule. *IEEE Transactions on Systems, Man and Cybernetics*, SMC-6(4):325–327, 1976.

[59] Nitin Bhatia and Vandana Ashev. Survey of nearest-neighbor techniques. *International Journal of Computer Science and Information Security*, 8(2): 302–305, 2010.

[60] Stephen M. Omohundro. Five balltree construction algorithms. Technical report, 1989.

[61] Jon Louis Bentley. Multidimensional binary search trees used for associative searching. *Commun. ACM*, 18(9):509–517, September 1975.

[62] C. D. Yu, J. Huang, W. Austin, B. Xiao, and G. Biros. Performance optimization for the k-nearest neighbors kernel on x86 architectures. In *SC15: International Conference for High Performance Computing, Networking, Storage and Analysis*, pages 1–12, 2015.

[63] R. Clark, G. Punzo, and M. Macdonald. Consensus speed optimisation with finite leadership perturbation in k-nearest neighbour networks. In

2016 IEEE 55th Conference on Decision and Control (CDC), pages 879–884, 2016.

[64] Leo Breiman, Jerome H Friedman, Richard A Olshen, and Charles J Stone. *Classification and regression trees*. The Wadsworth statistics probability series. Wadsworth & Brooks/Cole Advanced Books & Software, Monterey, CA, 1984.

[65] J. R. Quinlan. Induction of decision trees. *Machine Learning*, 1(1):81–106, Mar 1986.

[66] Lior Rokach and Oded Maimon. *Data mining with decision trees. Theory and applications*. World Scientific Pub Co Inc, 2008.

[67] Leo Breiman. Random Forests. 45:5–32, 2001.

[68] Jerome H Friedman. Greedy Function Approximation : A Gradient Boosting Machine. *The Annals of Statistics,*, 29(5):1189–1232, 2001.

[69] Jerome H Friedman. Stochastic gradient boosting. *Computational Statistics & Data Analysis - Nonlinear methods and data mining*, 38:367–378, 2002.

[70] Corrina Cortes and Vladimir Vapnik. Support-Vector Networks. *Machine learning*, 20:273–297, 1995.

[71] F. Rosenblatt. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, pages 65–386, 1958.

[72] J J Hopfield. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Sciences*, 79(April):2554–2558, 1982.

[73] Strother H Walker and David B Duncan. Trust Estimation of the Probability of an Event as a Function of Several Independent Variables. *Biometrika*, 54(1):167–179, 1967.

[74] M. E. Maron. Automatic indexing: An experimental inquiry. *J. ACM*, 8 (3):404–417, July 1961.